

Création d'Applications sur Internet

Chapitre5b: Langage PHP

Fatemeh BORRAN

Sommaire

- Transmettre des données de page en page
 - Transmettre des données avec l'URL
 - Transmettre des données avec les formulaires

Transmettre des données avec l'URL

- Exemple: comment chercher le mot `php` sur Google?
 1. Chercher avec l'interface de Google
 2. Chercher directement avec l'URL: `http://www.google.ch/search?q=php`
- Syntax:
`http://nom_site/nom_page?param1=val1¶m2=val2`
- Exemple:
`http://www.monsite.com/bonjour.php?nom=Dupont&prenom=Jean`

Exemple avec l'URL

Nous avons deux fichiers:

- index.html

```
<!DOCTYPE html>
<html>
  <head><title>Index</title></head>
  <body>
    <a href="bonjour.php?nom=Dupont&prenom=Jean">Dis-moi Bonjour!</a>
  </body>
</html>
```

Try now

- bonjour.php

```
<?php
echo "Bonjour " . $_GET['prenom'] . " " . $_GET['nom'] . "!";
?>
```



Exemple avec l'URL

Il faut toujours vérifier les paramètres:

- index.html

```
<!DOCTYPE html>
<html>
  <head><title>Index</title></head>
  <body>
    <a href="bonjour.php?nom=Dupont&prenom=Jean">Dis-moi Bonjour!</a>
  </body>
</html>
```

Try now

- bonjour.php

```
<?php
  if (isset($_GET['prenom']) AND isset($_GET['nom'])) {
    echo 'Bonjour ' . $_GET['prenom'] . ' ' . $_GET['nom'] . ' !';
  }
  else { // Il manque des paramètres, on avertit le visiteur
    echo 'Il faut renseigner un nom et un prénom !';
  }
?>
```

Transmettre des données avec l'URL

- Une URL représente l'adresse d'une page web (commençant généralement par `http://`)
- Lorsqu'on fait un lien vers une page, il est possible d'ajouter des paramètres sous la forme `nom_page.php?param1=val1¶m2=val2` qui seront transmis à la page.
- La page `nom_page.php` dans l'exemple précédent recevra ces paramètres dans un tableau nommé `$_GET`:
 - `$_GET['param1']` aura pour valeur `val1`.
 - `$_GET['param2']` aura pour valeur `val2`.
- Il faut pas faire confiance aux informations envoyé par l'URL, et il faut toujours tester prudemment leur valeur avant de les utiliser.

Transmettre des données avec les formulaires

Rappel sur les formulaires:

- Les formulaires sont des éléments HTML permettant de récupérer des données auprès des visiteurs d'une page Web. Ils sont délimités par les balises `<form>` et `</form>` (voir Chap3c).
- L'attribut `action` permet de spécifier la page vers laquelle les données doivent être envoyées.
- L'attribut `method` permet de spécifier si les données doivent être encodées dans l'URL (`method="GET"`) ou dans le corps de la requête HTTP (`method="POST"`).
- On attribue un nom à chaque élément du formulaire à l'aide de leur attribut `name`.

Récupérer des données

- PHP se charge de récupérer les données envoyées depuis un formulaire et de les placer dans une variable **superglobale** (variable un peu particulière propre au langage PHP).
- Selon que la méthode du formulaire utilisée est **POST** ou **GET**, l'accès à une valeur d'un champ de formulaire nommé `nomChamp` s'écrit comme suit:
 - `$valeur = $_POST['nomChamp'];`
 - ou
 - `$valeur = $_GET['nomChamp'];`

Exemple avec Formulaire

Code d'un formulaire HTML basique, form.html:

```
<!DOCTYPE html>
<html>
  <head><meta charset="utf-8"><title>Formulaire</title></head>
  <body>
    <form action="myPage.php" method="GET">
      <table>
        <tr>
          <td>Nom : </td><td><input type="text" name="name"/></td>
        </tr>
        <tr>
          <td>Prénom : </td><td><input type="text" name="surname"/></td>
        </tr>
        <tr>
          <td></td><td><input type="submit"/></td>
        </tr>
      </table>
    </form>
  </body>
</html>
```

Try now

Les données envoyé par la méthode GET dans l'URL:

`http://localhost/CAI/myPage.php?name=Borran&surname=Fateme`

Exemple avec Formulaire (page PHP)

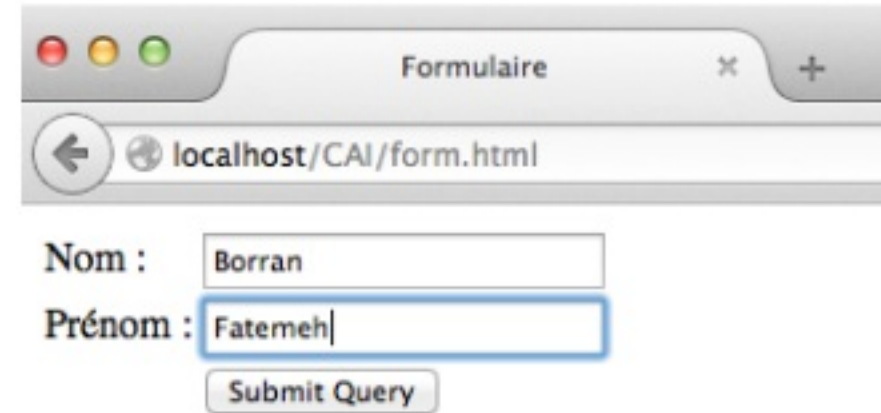
Code de la page PHP chargée de récupérer les données du formulaire et de les afficher, myPage.php:

```
<?php
    $nom      = $_GET['name'];
    $prenom   = $_GET['surname'];
?>
<html>
    <head>
        <title>Ma page</title>
    </head>
    <body>
        <h1>Bonjour <?php echo $prenom; ?> !</h1>
        <p>Ton nom de famille est <strong><?php echo $nom; ?></strong></p>
    </body>
</html>
```

Try now

Résultat de l'exemple

form.html



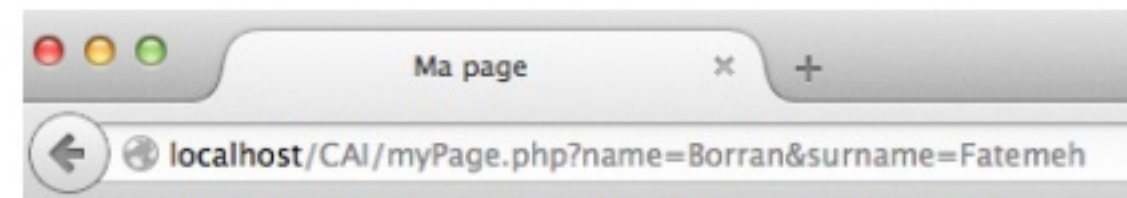
Formulaire

localhost/CAI/form.html

Nom :

Prénom :

myPage.php



Ma page

localhost/CAI/myPage.php?name=Borran&surname=Fateme'h

Bonjour Fateme'h !

Ton nom de famille est **Borran**

POST ou GET?

- La méthode `POST` est utilisée pour transmettre des données via un formulaire et passer en "caché" (via une deuxième requête HTTP, après la première qui demande la page html/php), on ne voit pas les données des variables explicitement.
- Contrairement à `POST`, la méthode `GET` passe les variables via l'URL (en une seule requête HTTP), les données sont donc "visible" dans l'adresse de la page.
- *Attention:* un URL est limitée à 255 caractères!

Variables superglobales

- Elles sont écrites en majuscules et commencent toutes par un underscore _.
- Elles sont des tableaux car elles contiennent généralement de nombreuses informations.
- Elles sont automatiquement créées par PHP à chaque fois qu'une page est chargée. Elles existent donc sur toutes les pages et sont accessibles partout : au milieu de votre code, au début, dans les fonctions, etc.
- Exemple: Le code `$_SERVER['REMOTE_ADDR']` nous donne l'adresse IP du client qui a demandé à voir la page, ce qui peut être utile pour l'identifier.

Sessions

- Les sessions constituent un moyen de conserver/stocker des variables sur le serveur le temps de la présence d'un visiteur sur toutes les pages de votre site.
- Ceci ne serait pas facile avec `GET` et `POST` car ils sont plutôt faits pour transmettre les informations une seule fois, d'une page à une autre.
- Mais alors comment se fait-il que, lorsque je me connecte à ma boîte aux lettres Hotmail, le site ne me demande pas un mot de passe à chaque clic?
- Comment Hotmail sait-il qu'il s'agit de requêtes HTTP envoyées par moi-même?
- Ecriture et lecture d'une variable session:

```
$_SESSION['prenom'] = "Fateme";  
$prenom = $_SESSION['prenom'];
```


Fonctions à connaître

- Dans une page PHP, pour pouvoir utiliser les variables de session, il faut **toujours** commencer par invoquer la méthode `session_start()`.
- `session_start()`: démarre le système de sessions. Si le visiteur vient d'arriver sur le site, alors un numéro de session est généré pour lui. Vous devez appeler cette fonction au tout début de chacune des pages où vous avez besoin des variables de session.
- `session_destroy()`: ferme la session du visiteur. Cette fonction est automatiquement appelée lorsque le visiteur ne charge plus de page de votre site pendant plusieurs minutes (c'est le **timeout**), mais vous pouvez aussi créer une page **Déconnexion** si le visiteur souhaite se déconnecter manuellement.

Exemple (création de session)

Code d'une page PHP chargée de créer deux variables de session, `saveSession.php`:

```
<?php
    session_start();
    $_SESSION['nom']      = "Borran";
    $_SESSION['prenom'] = "Fateme";
?>
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Créer une session</title>
    </head>
    <body>
        <strong> Session créée ! </strong>
        <a href="restoreSession.php"> Autre page </a>
    </body>
</html>
```

Try now

Exemple (récupération des valeurs)

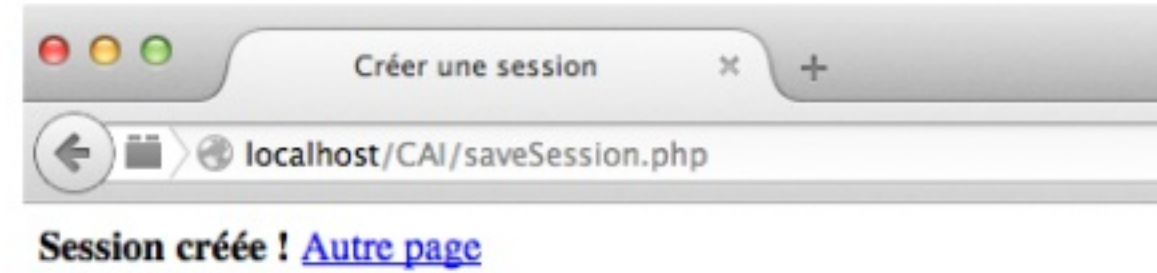
Code de la page PHP chargée de récupérer les valeurs des deux variables de session, `restoreSession.php`:

```
<?php
    session_start();
    $nom      = $_SESSION['nom'];
    $prenom  = $_SESSION['prenom'];
?>
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Restaurer une session</title>
    </head>
    <body>
        <strong> Contenu de la session : </strong><br/><br/>
        Nom : <strong><?php echo($nom); ?></strong><br/>
        Prenom : <strong><?php echo($prenom); ?></strong>
    </body>
</html>
```

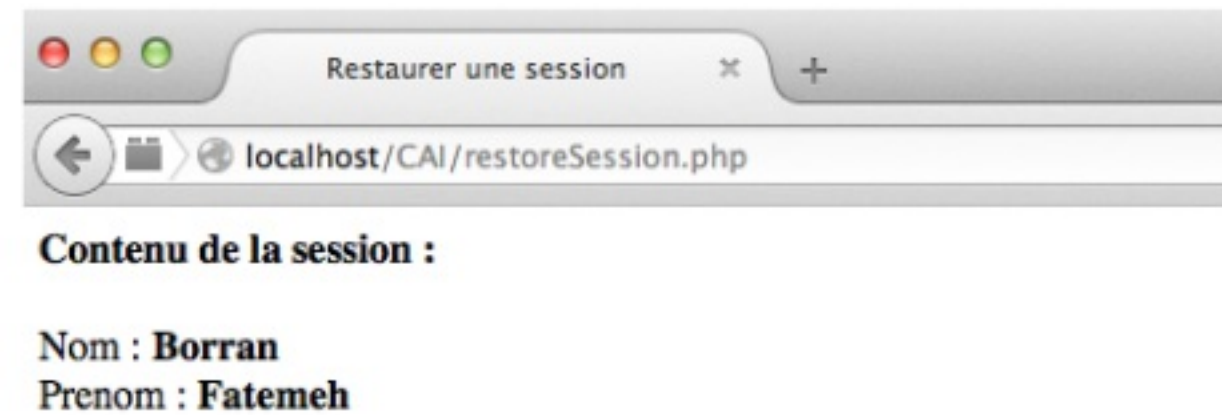
Try now

Exemple (utilisation)

saveSession.php



restoreSession.php



L'utilité des sessions

- Imaginez un script qui demande un login et un mot de passe pour qu'un visiteur puisse se "connecter" (s'authentifier). On peut enregistrer ces informations dans des variables de session et se souvenir de l'identifiant du visiteur sur toutes les pages du site!
- Puisqu'on retient son login et que la variable de session n'est créée que s'il réussit à s'authentifier, on peut l'utiliser pour restreindre certaines pages de notre site à certains visiteurs uniquement. Cela permet de créer toute une zone d'administration sécurisée : si la variable de session login existe, on affiche le contenu, sinon on affiche une erreur.
- On se sert activement des sessions sur les sites de vente en ligne. Cela permet de gérer un "panier": on retient les produits que commande le client quelle que soit la page où il est. Lorsqu'il valide sa commande, on récupère ces informations et... on le fait payer.

exemple.php (1/2)

```
<?php
session_start();
if (!isset($_POST['nom']) OR !isset($_POST['prenom'])
    OR !isset($_POST['sexe'])) {
    if (!isset($_SESSION['nbr'])) {
        $_SESSION['nbr'] = 0;
    }
    else {
        $_SESSION['nbr'] = $_SESSION['nbr'] + 1;
        echo "Cela fait " . $_SESSION['nbr'] .
            " fois que tous les champs ne sont pas remplis!<br/>";
    }
}
else {
    echo "Bonjour " . $_POST['prenom'] . " " . $_POST['nom'] . " <br/>";
    $result = "";
    if ($_POST['sexe'] == "F") {
        $result = "une femme";
    }
    else {
        $result = "un homme";
    }
    echo "Vous êtes {$result}";
    session_destroy();
}
?>
```


exemple.php (2/2)

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Bonjour</title>
  </head>
  <body>
    <form method="post" action="exemple.php">
      <label> Nom : </label>
      <input type="text" name="nom"/><br/>
      <label> Prenom : </label>
      <input type="text" name="prenom"/><br/>
      <label> Sexe :</label>
      <input type="radio" name="sexe" value="F"/>Femme
      <input type="radio" name="sexe" value="H"/>Homme<br/>
      <input type="submit" value="Envoyer"/>
    </form>
  </body>
</html>
```

Try now

Questions?!

